1. **Write an ALP to add two 8-bit numbers.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Num1 db 2

Num2 db 3

.code segment

main proc

mov ax, @data

mov ds,ax

mov ch, num1

mov dh, num2

print “your addition is :”

add ch,dh

add ch,48

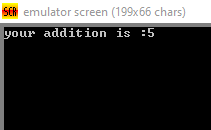
mov dl,ch

mov ah,02h

int 21h

main endp

end main



1. **Write an ALP to subtract two 8-bit numbers.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Num1 db 7

Num2 db 3

.code segment

main proc

mov ax, @data

mov ds,ax

mov ch, num1

mov dh, num2

print “your subtraction is :”

sub ch,dh

sub ch,48

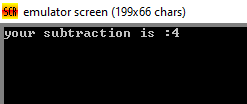
mov dl,ch

mov ah,02h

int 21h

main endp

end main



1. **Write an ALP to multiply two 8-bit numbers.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Num1 db 2

Num2 db 3

.code segment

main proc

mov ax, @data

mov ds,ax

print “your multiplication is :”

mov ah,0

mov al, num1

mov bl, num2

mul bl

mov dl,al

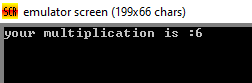
add dl,48

mov ah,02h

int 21h

main endp

end main



1. **Write an ALP to divide two 8-bit numbers.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Num1 db 8

Num2 db 4

.code segment

main proc

mov ax, @data

mov ds,ax

print “your division is :”

mov ah,0

mov al, num1

mov bl, num2

div bl

mov dl,al

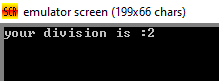
add dl,48

mov ah,02h

int 21h

main endp

end main



1. **Write an ALP to Add two 16-bit numbers.**

include ‘emu8086.inc’

Data segment

Num1 dw 4848

Num2 dw 3

sum dw ?

Data ends

Code segment

Start:

Mov ax,data

Mov ds,ax

Mov ax,num1

add num2

Mov sum,ax

Call print\_num

hlt

Define\_print\_num

Define\_print\_num\_uns

Code ends

End start

End



1. **Write an ALP to subtract two 16-bit numbers.**

include ‘emu8086.inc’

Data segment

Num1 dw 4848

Num2 dw 1111

Dif dw ?

Data ends

Code segment

Start:

Mov ax,data

Mov ds,ax

Mov ax,num1

sub num2

Mov dif,ax

Call print\_num

hlt

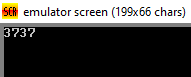
Define\_print\_num

Define\_print\_num\_uns

Code ends

End start

End



1. **Write an ALP to multiply two 16-bit numbers.**

include ‘emu8086.inc’

Data segment

Num1 dw 5

Num2 dw 9

Pro dw ?

Data ends

Code segment

Start:

Mov ax,data

Mov ds,ax

Mov ax,num1

Mul num2

Mov pro,ax

Call print\_num

hlt

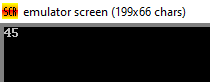
Define\_print\_num

Define\_print\_num\_uns

Code ends

End start

End



1. **Write an ALP to divide two 16-bit numbers.**

include ‘emu8086.inc’

Data segment

Num1 dw 2222

Num2 dw 2

D dw ?

Data ends

Code segment

Start:

Mov ax,data

Mov ds,ax

Mov ax,num1

div num2

Mov D,ax

Call print\_num

hlt

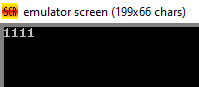
Define\_print\_num

Define\_print\_num\_uns

Code ends

End start

End



1. **Write an ALP to calculate square of a numbers.**

include ‘emu8086.inc’

.Data

Num1 db 9h

Sqr dw 1 dup()

.Code segment

Main proc

Mov ax,@data

Mov ds,ax

Mov ah,0

Mov al,num1

Mul al

Mov sqr,ax

Call print\_num

hlt

mainendp

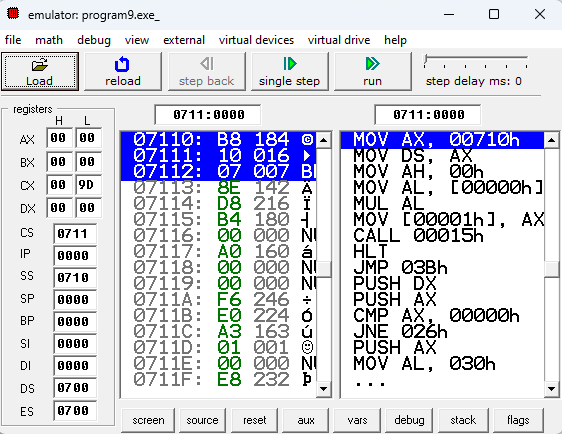
define\_print\_num

Define\_print\_num\_uns

Code ends

End main

Output:-



**AIM:-Write an ALP to calculate factorial of 16-bit number.**

Include ‘emu8086.inc’

Mov ax,07h

Mov cx,ax

Dec cx

Back:mul cx

Loop back

Call\_print\_num

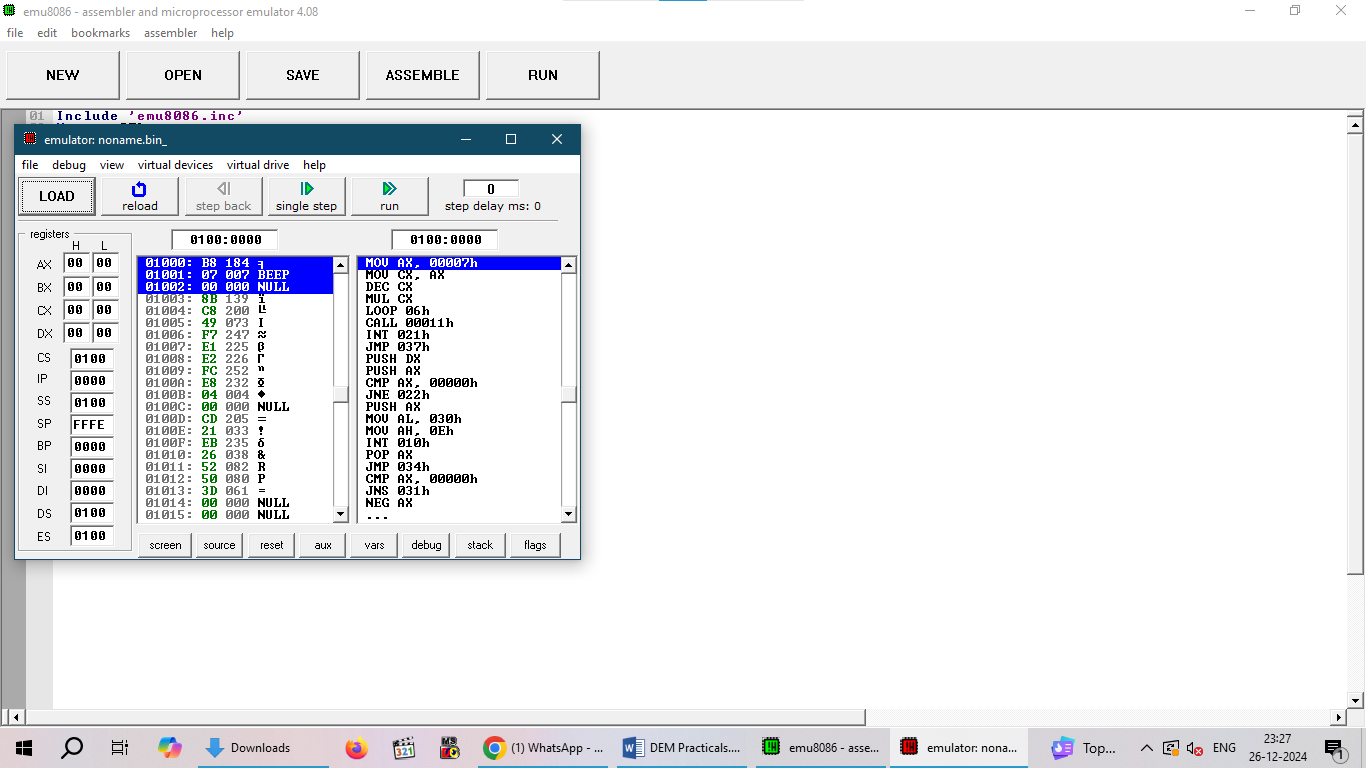
Int 21h

Define\_print\_num

Define\_print\_num\_uns

End

Output:-



**AIM:-Write an ALP to print an array.**

include ‘emu8086.inc’

.model small

.stack 100h

.data

Array db 3,4,5,6,7

Data ends

.code

Main proc

Mov ax,@data

Mov ds,ax

Mov si,offset array

Mov cx,5

Print “the array values are:”

Repeate:

Mov dl,[si]

Add dl,48

Mov ah,02h

Int 21h

Mov dl,32

Mov ah,02h

Int 21h

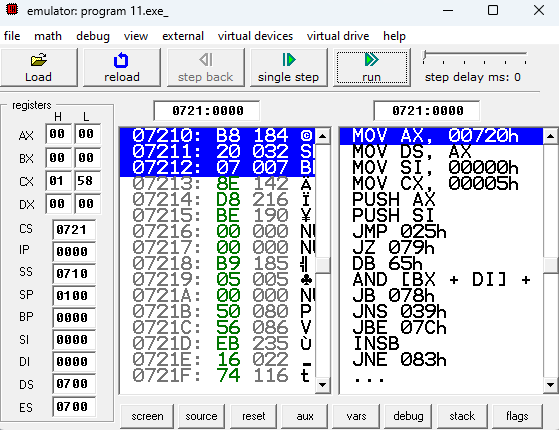
Inc si

Loop repeat

Main endp

end main

output:-



**AIM:-Write an ALP to sort an array.**

include ‘emu8086.inc’

.model small

.stack 100h

.data

arr db 23,56,45,32

data ends

.code segment

Mov ax,@data

Mov ds,ax

Mov ch,04h

Up2 : mov cl,04h

Lea si,arr

Up1 : mov al,[si]

Mov bl,[si+1]

Cmp al, bl

Jc down

Mov dl,[si+1]

Xchg [si],dl

Mov [si+1],dl

Down: inc si

Dec cl

Jnz up1

Dec ch

Jnz up2

Lea si,arr

Mov ch,05h

Mov ah,0

Again:

Mov al,[si]

Call print\_num

Print ‘ ’

Inc si

Dec ch

Jnz again

Hlt

Define\_print\_num

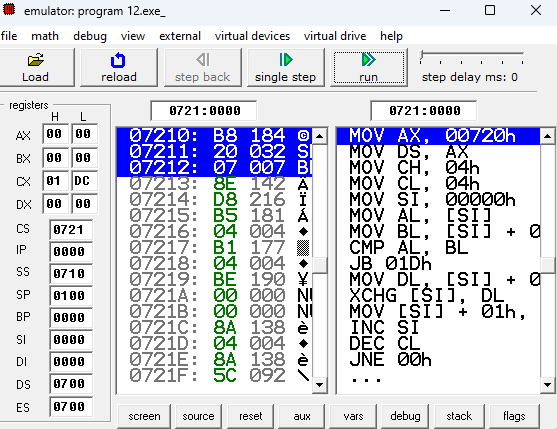
Define\_print\_num\_uns

Code ends

End

End main

Output:-



**AIM:-Write an ALP to calculate sum of array.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Arr db 1,2,3,4,5

Sum db 0

.code

main proc

mov ax,@data

mov ds,ax

mov cx,5

mov ax,0

mov bx,offset arr

repeat:

add al,[bx]

inc bx

dec cx

jnz repeat

mov sum,al

mov dl,sum

call print\_num

int 21h

define\_print\_num

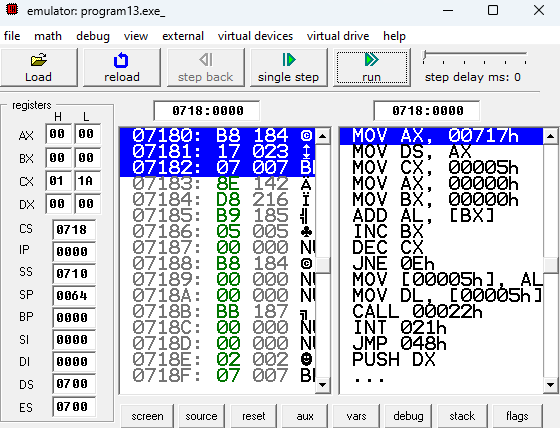
define\_print\_num\_uns

end

main endp

end main

output:-



**AIM:-Write an ALP to move block of data overlap.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Blk1 db 01,02,03,04,05,06,07,08,09,0ah

Blk2 db 10 dup()

.code

mov ax,@data

mov ds,ax

mov es,ax

mov si,offset blk1

mov di,offset blk2

mov cx,0ah

add si,0009h

add di,0004h

again:

mov al,[si]

mov [di],al

dec si

dec di

dec cl

jnz again

mov ah,4ch

call print\_num

int 21h

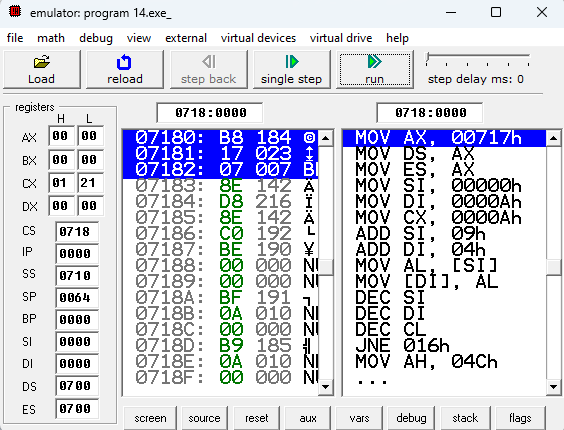
define\_print\_num

define\_print\_num\_uns

end

end main

output:-



**AIM:-Write an ALP TO MOVE BLOCK OF DATA WITHOUT OVERLAP.**

Include ‘emu8086.inc’

.model small

.stack 100

.data

Blk1 db 01,02,03,04,05,06,07,08,09,0ah

Blk2 db 10 dup()

.code

mov ax,@data

mov ds,ax

mov es,ax

mov si,offset blk1

mov di,offset blk2

mov cx,10

again:cld

rep movsb

lea si,blk2

mov ch,0ah

mov ah,0

n:

mov al,[si]

call print\_num

print

inc si

dec chc

jnz n

hlt

define\_print\_num

define\_print\_num\_uns

end

hlt

end main

\

Output:-

